UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/081,000	02/20/2002	Anatoli Fomenko	P-6507	1328

24209 7590 01/03/2006

GUNNISON MCKAY & HODGSON, LLP
1900 GARDEN ROAD
SUITE 220
MONTEREY, CA 93940

EXAMINER	
LEROUX, ETIENNE PIERRE	
ART UNIT	**PAPER NUMBER**
2161	

DATE MAILED: 01/03/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	10/081,000	FOMENKO, ANATOLI	
	Examiner	Art Unit	
	Etienne P. LeRoux	2161	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *24 October 2005*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-32* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-32* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *20 February 2002* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

Claim Status

Claims 1-32 are pending. Claims 1-32 are rejected as detailed below.

Claim Rejections - 35 USC § 102

The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

Claims 1-14 and 17-31 are rejected under 35 U.S.C. 102(e) as being anticipated by Pub No US 2002/0116702 issued to Aptus et al (hereafter Aptus).

Claim 1:

Aptus discloses:

main interfaces defining versioning functionality, said main interfaces allowing access to the versioning functionality [Fig 20, 610, paragraph 89]

a functional implementation of said main interfaces, said functional implementation comprising classes [paragraph 93] and libraries [Fig 9, paragraph 79] implementing the versioning functionality, said classes including a reference to a program module to perform a requested versioning function [paragraph 93], and

a user interface for using the versioning functionality [Fig 20, 2002, 2004, 2006, Fig 21, 2102, paragraphs 89, 92 and 93].

Claim 2:

Aptus discloses a communication mechanism implementing client-server functionality [paragraph 89, Fig 20, 2012, 2014]

Claims 3 and 23:

Aptus discloses:

an interface defining versioning server functionality [Fig 20, 2014],

an interface defining versioning client functionality [Fig 20, 2012],

an interface defining versioning repository functionality [Fig 20, 2016],

an interface defining designated directory structures and access to the designated directory structures [paragraph 89, Fig 20, 2007]; and

an interface defining transactions between the designated directory structures [paragraph 89, Fig 20, 2007]

Claims 4 and 24:

Aptus discloses an interface defining file actions within a designated directory structure [paragraph 89]

Claim 5:

Aptus discloses native programming interfaces allowing code written in the object-oriented platform-independent language to operate with code written in a native language [Fig 2, 202, paragraph 48] other than the object-oriented platform-independent language [Fig 2, 200, paragraph 48]

Claim 6:

Aptus discloses wherein said functional implementation comprises classes and first libraries written in an object-oriented platform-independent programming language [Fig 3, 300, Fig 3, 302, paragraph 50]; and second libraries including software routines written in a native programming language other than the object-oriented platform-independent language, said second libraries implementing said native programming interfaces [Fig 9, paragraph 79]

Claim 7:

Aptus discloses a reference to a first library, said reference being invoked if a requested versioning function is implemented with the object-oriented platform-independent programming language [Fig 3, 300, Fig 3, 302, paragraph 50], and a reference to a native function and a second library, said reference being invoked, using a native programming interface, if a requested versioning function is implemented with the native programming language [Fig 9, paragraph 79]

Claims 8 and 25:

Aptus discloses resource files available to said classes and libraries [paragraph 79]

Claims 9 and 26:

Aptus discloses a class BringoverFrom including a reference to a program module for copying master files stored in a first directory structure and thereby creating a set of working files, and a class BringoverTo including a reference to a program module for storing the set of working files in a second directory structure [paragraph 94]

Claims 10 and 27:

Aptus discloses a class PutbackFrom including a reference to a program module for copying the working files in the second directory stlucture and thereby creating a set of updated files; and a class PutbackTo including a reference to a program module for replacing a

corresponding set of the master files in the first directory structure with the set of updated files [paragraph 94].

Claims 11 and 28:

Aptus discloses a reference to a program module for receiving a request for replacing the master files with a set of updated files, and checking for a previous replacement of the master files with another set of updated files, a reference to said class PutbackTo; a reference to said class BringoverFrom; and a reference to said class BringoverTo [Fig 9, paragraph 79].

Claim 12:

Aptus discloses wherein said reference to said class Putbackto is invoked if there is no previous replacement of the master files [paragraph 93]

Claim 13:

Aptus discloses wherein said reference to said class Bringoverfrom and said reference to said class Bringoverto are invoked if there is a previous replacement of the master files [paragraph 93]

Claim 14:

Aptus discloses a reference to a program module for creating a writeable copy of a working file stored in the second directory structure; and a reference to a program module for storing the writeable copy to a requested address; and a class Checkin including a reference to a program module for copying the writeable copy of a requested address so as to create an updated working file; and a reference to a program module for replacing the working file with the updated working file [Fig 7, paragraph 54].

Claim 17:

Apt discloses a graphic user interface and a command line interface [Fig 21].

Claim 18:

Pâté discloses defining versioning functionality in main intedàces of said versioning API [fig 20, 610, paragraph 89]; implementing the versioning functionality in classes [paragraph 93] and libraries [Fig 9, paragraph 79] of said versioning API, the librades including first libraries written in an object-oriented platform-independent progrnmming language [Fig 2, 200, paragraph 48], and second libraries written in a native progrnmming language [Fig 2, 202, paragraph 48] other than the object-oriented platform-independent language, and providing native programming interfaces allowing code written in the object-oriented platform-independent language to operate with code written in a native language other than the object-oriented platform-independent language, the second libraries including native progrnmming interface implementation [Fig 2, 204, 206]

Claim 19:

Aptus discloses receiving, from a client, a request for a versioning function, calling a class implementing the requested versioning function; invoking a first library from the class, if the requested versioning function is implemented in the first library written in the object-oriented platform-independent program language; and using a native programming interface from the class, so as to invoke a second library if a requested versioning function is implemented in the second library written in a native language other than the object-oriented platform-independent language [paragraph 80].

Claim 20:

Aptus discloses making a call for a method of a proxy object at the client, the proxy object being associated with a type of versioning transaction; converting the call for a method to a request of the method; transmitting the request to the hosting server; and invoking a servlet at the hosting server to generate a response to the request, the servlet delegating processing of the request to a sen'er object calling a class including the requested method; involdng, from the class, the method directly if the requested method is implemented in a first library written in the object-oriented platform-independent program language; and invoking, from the class, the method using a native proçamming interface if the requested method is implemented in a second library written in a native language other than the object-oriented platform-independent progrnm language [paragraph 94].

Claim 21:

Aptus discloses making a call with a graphic user interface [Fig 21, 2102]

Claim 22:

Aptus discloses making a call with a command line interface [Fig 21, 2002]

Claim 29:

Aptus discloses creating a writeable copy of a working file stored in the second directory structure and storing the writeable copy to a requested address [paragraph 79, Fig 9]

Claim 30:

Aptus discloses copying the writeable copy so as to create an updated working file, and replacing the working file in the second directory with the updated working file [Fig 9, paragraph 79]

Claim 31:

Aptus discloses receive a request fro creating a writable copy of a working file, and checking whether a writeable copy of the working file has already been created [paragraph 79, Fig 9]

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

Claim 15 is rejected under 35 U.S.C. 103(a) as being unpatentable over Aptus in view of US Pat No 6,018,743 issued to Xu (hereafter Xu).

Claim 15:

Aptus discloses the elements of claims 1, 9 and 10, as noted above but is silent regarding a class lock including a reference to a program module for receive a request for creating as writeable copy of a working files and checking whether a writeable copy of the working file has already been created. Xu discloses a lock mode to define locks such as read locks and write locks [col 21, lines 13-25]. It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Aptus to include a class lock including a reference to a program module for receive a request for creating as writeable copy of a working files and checking whether a writeable copy of the working file has already been created based on the teaching of Xu for the purpose of controlling the updating of the master file such that the integrity of the master filed is maintained at all times.

Claims 16 and 32 are rejected under 35 U.S.C. 103(a) as being unpatentable over Aptus in view of US Pat No 6,681,382 issued to Kakumani et al (hereafter Kakumani).

Claims 16 and 32:

Aptus discloses the elements of claims 1 and 9 as noted above but is silent regarding a class Freezepoint including a reference to a program for creating freezepoint files for files in a specified directory structure, the freezepoint files storing a specific time stamp and then current version of the corresponding files. Kakumani discloses a file including time stamp and version number [col 2, lines 13-20]. It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Aptus to include a class Freezepoint including a reference to a program for creating freezepoint files for files in a specified directory structure, the freezepoint files storing a specific time stamp and then current version of the corresponding files based on the teaching of Kakumani for the purpose of designating centrally stored files such that they can be easily retrieved [col 2, lines 13-16].

Response to Arguments

Applicant's arguments filed 10/24/2005 have been fully considered but they are not persuasive.

Applicant Argues:

Applicant states in the second and third paragraphs of page 13 "This paragraph describes generally a version control system and describes only a more intuitive interface. It does not describe how this interface is implemented, and does not use interface as recited in Claim 1.

Claim 1 does not recite just some interface, but rather a particular type of interface. In particular Applicant defined:

An interface is a named collection of method definitions and defines a protocol of behavior that can be implemented by any class in the class hierarchy. An interface defines a set of method[s] but does not implement them.
Specification paragraph 21.

Examiner Responds:

Examiner is not persuaded. Examiner was unable to find above definition in paragraph 21 of the specification. Furthermore, MPEP § 2106 requires Office personnel to give claims their broadest reasonable interpretation in light of the supporting disclosure. In re Morris, 127 F.3d 1048, 1054-55, 44 USPQ2d 1023, 1027-28 (Fed. Cir. 1997). Limitations appearing in the specification but not recited in the claim are not read into the claim. E-Pass Techs., Inc. v. 3Com Corp., 343 F.3d 1364, 1369, 67 USPQ2d 1947, 1950 (Fed cir 2003) (claims must be interpreted in view of the specification without importing limitations from the specification into the claims unnecessarily). In re Prater, 415 F.2d 1393, 1404-05, 162 USPQ 541, 550-551(CCPA 1969). See also In re Zletz, 893 F.2d 319, 321-22, 13 USPQ2d 1320, 1322(Fed. Cir. 1989) (During patent examination the pending claims must be interpreted as broadly as their terms reasonable allow .... The reason is simply that during patent prosecution when claims can be amended, ambiguities should be recognized, scope and breadth of language explored, and clarification imposed .... An essential purpose of patent examination is to fashion claims that are precise, clear, correct and unambiguous. Only in this way can uncertainties of claim scope be removed, as much as possible, during the administrative process.

The MPEP, as recited above, requires examiners to give claims their broadest reasonable interpretation in light of the supporting disclosure. A reasonable interpretation of "interface" is

the following: Software that enables a program to work with the user (the user interface, which can be a command-line interface, menu-driven interface, or a graphical user interface), with another program such as the operating system, or with the computer's hardware.[1] In line with above interpretation, Aptus discloses the following in paragraph 89:

[0089] In addition to the functionality described above, the improved software development tool integrates a version control system that permits programmers using different computers to work simultaneously on a software project by managing the various versions of the source code associated with the software project. The improved software development tool also enables programmers to interact with the version control system by manipulating a diagram or diagram element associated with a software project, thus facilitating the use of the version control system through a more intuitive interface and a more natural grouping of files. For example, FIG. 20 depicts data processing system 2000, which includes a number of computers 2002-2008 connected via a network 2010, where the users of the computers are using the version control system of the improved software development tool 610. On computers 2002-2006, software development tool 610 includes a client component 2012 of the version control system. On computer 2008, the software development tool 610 contains a server component 2014 of the version control system. Computer 2008 is pre-designated as containing a central repository 2016. Central repository 2016 is a shared directory for storing a master copy of project 612. Project 612 comprises all of the source files in a particular software project. Each of the computers 2002-2006 also includes a working directory 2007 that contains working copies of source files that programmers can make changes to without affecting the master copy in the central repository 2016.

Aptus discloses a software development tool with version control which interfaces with programmers using computers 2002-2006 and which also interfaces with multiple versions of the source code being developed for the project. Aptus also discloses programmers manipulate a diagram and thus Aptus discloses a graphical user interface. Examiner correctly maintains the disclosure of Aptus regarding "interface" is in accord with the definition in the Microsoft Computer Dictionary.

Furthermore, examiner maintains the above disclosure by Aptus reads on the language of the **claim limitation** (emphasis added), i.e., claim 1, which states "main interfaces defining

[1] Microsoft Computer Dictionary, Fifth Edition

version functionality, said main interfaces allowing access to the versioning software" because of the above multiple references to the software development tool which interfaces with multiple versions of the source code. Further, the software development tool interfaces with a server which comprises a central repository which in turn interfaces with programmers using client computers 2002, 2004 and 2006.

Still further, consider the claimed "interface" which is further claimed as follows in claim 1, "a functional implementation of said main interfaces, said functional implementation comprising classes and libraries implementing the versioning functionality." Examiner maintains the following disclosure by Aptus reads on above claim limitation.

Aptus paragraph 79:

0079] FIG. 9 depicts a flow diagram of the steps performed by the software development tool to develop a project in accordance with methods consistent with the present invention. As previously stated, the project comprises a plurality of files. The developer either uses the software development tool to open a file that contains existing source code, or to create a file in which the source code will be developed. If the software development tool is used to open the file, determined in step 900, the software development tool initially determines the programming language in which the code is written (step 902). The language is identified by the extension of the file, e.g., ".java" identifies source code written in the Java.TM. language, while ".cpp" identifies source code written in C++. The software development tool then obtains a template for the current programming language, i.e., a collection of generalized definitions for the particular language that can be used to build the data structure (step 904). For example, the templates used to define a new Java.TM. class contains a default name, e.g., "Class 1," and the default code, "public class Class1 { }." Such templates are well known in the art. For example, the "Microsoft Foundation Class Library" and the "Microsoft Word Template For Business Use Case Modeling" are examples of standard template libraries from which programmers can choose individual template classes. The software development tool uses the template to parse the source code (step 906), and create the data structure (step 908). After creating the data structure or if there is no existing code, the software development tool awaits an event, i.e., a modification or addition to the source code by the developer (step 910). If an event is received and the event is to close the file (step 912), the file is saved (step 914) and closed (step 916). Otherwise, the software development tool performs the event (step 918), i.e., the tool makes the modification. The software development tool then updates the TMM or model (step 920), as discussed in detail below, and updates both the graphical and the textual views (step 922).

Aptus per the above, clearly discloses libraries from which programmers can choose individual template classes. The above teaching can be accurately mapped to the claim limitation "a functional implementation of said main interfaces, said functional implementation comprising classes and libraries implementing the versioning functionality"

Conclusion

THIS ACTION IS MADE FINAL. Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Contact Information

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Etienne P. LeRoux whose telephone number is (571) 272-4022. The examiner can normally be reached Monday through Friday, 8:00am – 4:30pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Safet Metjahic can be reached on (571) 272-4023. The fax phone number for the organization where this application or proceeding is assigned is (571) 273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Etienne LeRoux

12/28/2005